# NAVAL POSTGRADUATE SCHOOL
## Monterey, California

# THESIS

**APPLICATION OF THE NOGUEIRA RISK ASSESSMENT MODEL TO REAL-TIME EMBEDDED SOFTWARE PROJECTS**

by

Craig S. Johnson
Robert A. Piirainen

June 2001

| | |
|---|---|
| Thesis Advisor: | Luqi |
| Second Reader: | Valdis Berzins |

20011116 168

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE<br>June 2001 | 3. REPORT TYPE AND DATES COVERED<br>Master's Thesis |
|---|---|---|

| 4. TITLE AND SUBTITLE<br>Application of the Nogueira Risk Assessment Model to Real-Time Embedded Software Projects | 5. FUNDING NUMBERS |
|---|---|

**6. AUTHOR(S)**
Johnson, Craig S. and Piirainen, Robert A.

| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Naval Postgraduate School<br>Monterey, CA 93943-5000 | 8. PERFORMING ORGANIZATION REPORT NUMBER |
|---|---|

| 9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSORING / MONITORING AGENCY REPORT NUMBER |
|---|---|

**11. SUPPLEMENTARY NOTES**

The views expressed in this Thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

| 12a. DISTRIBUTION / AVAILABILITY STATEMENT<br>Approved for public release, distribution is unlimited. | 12b. DISTRIBUTION CODE |
|---|---|

**13. ABSTRACT**

This Thesis addresses the application of a formal model for risk assessment to real-time embedded software development projects. It specifically targets the use of existing military and defense software development projects as a way to validate, or refine the formal model. In this case the Nogueira model. Data will be gathered from real projects and analyze through use of the Nogueira model. Selected projects were based on specific criteria, listed later in this Thesis. This is, in essence, a "post mortem" of these projects. It gives the ability to compare the model's predictions against what the real data collected from the projects indicated. Results will be reported with our conclusions as to the model's viability for use in determining risk as to probability of completion given the time allowed for the projects. These are data points in the validation of the model and the results, good or bad, cannot be used as a definitive substantiation of the model's fitness for use on other real projects.

| 14. SUBJECT TERMS<br>Requirements Volatility (RV), Change Rate (CR), Birth Rate (BR), Death Rate (DR), Complexity (CX), Large Granularity Complexity (LGC), Operators, Data Streams, Abstract Data Types (ADT), Efficiency Factor (EF), Software Engineering, Risk Assessment, Estimation Models, Bi-Dimensional Plot, Slim, Putnam, Function Points, COCOMO, Boehm, Prototype System Description Language (PSDL), Computer Aided Prototyping System (CAPS), Weibull Distribution | 15. NUMBER OF PAGES<br>75 |
|---|---|
| | 16. PRICE CODE |

| 17. SECURITY CLASSIFICATION OF REPORT<br>Unclassified | 18. SECURITY CLASSIFICATION OF THIS PAGE<br>Unclassified | 19. SECURITY CLASSIFICATION OF ABSTRACT<br>Unclassified | 20. LIMITATION OF ABSTRACT<br>UL |
|---|---|---|---|

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

i

# APPLICATION OF THE NOGUEIRA RISK ASSESSMENT MODEL TO REAL-TIME EMBEDDED SOFTWARE PROJECTS

Craig S. Johnson
Senior Computer Specialist
Defense Contract Management Agency
B.S.I.S., University of Phoenix, Arizona, 1999

Submitted in partial fulfillment of the
requirements for the degree of

## MASTER OF SCIENCE IN SOFTWARE ENGINEERING

Robert A. Piirainen
General Engineer
Tank Automotive Research and Development Command, TARDEC
B.S.M.E., Michigan Technological University, Michigan, 1973

Submitted in partial fulfillment of the
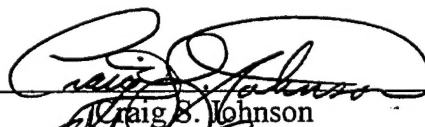requirements for the degree of
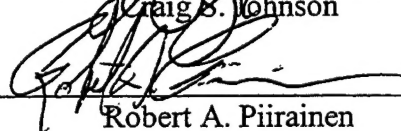
## MASTER OF SCIENCE IN SOFTWARE ENGINEERING
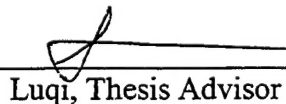
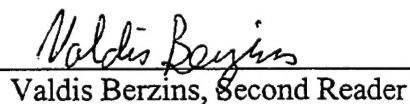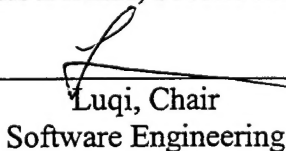from the

## NAVAL POSTGRADUATE SCHOOL
**June 2001**

Co-Authors: _____
Craig S. Johnson

_____
Robert A. Piirainen

Approved by: _____
Luqi, Thesis Advisor

_____
Valdis Berzins, Second Reader

_____
Luqi, Chair
Software Engineering

iii

# ABSTRACT

This Thesis addresses the application of a formal model for risk assessment to real-time embedded software development projects. It specifically targets the use of existing military and defense software development projects as a way to validate, or refine the formal model. In this case the Nogueira model.[1] Data will be gathered from real projects and analyzed through use of the Nogueira model. Selected projects were based on specific criteria, listed later in this Thesis. This is, in essence, a "post mortem" of these projects. It gives the ability to compare the model's predictions against what the real data collected from the projects indicated. Results will be reported with our conclusions as to the model's viability for use in determining risk as to probability of completion given the time allowed for the projects. These are data points in the validation of the model and the results, good or bad, cannot be used as a definitive substantiation of the model's fitness for use on other real projects.

---

[1] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," <u>Software Engineering Department Naval Postgraduate School Ph.D. Dissertation</u>, September 2000: 2

# TABLE OF CONTENTS

# LIST OF FIGURES

# ACKNOWLEGEMENTS

## Craig S. Johnson

To my family: two boys, Erik B. Johnson and Kyle B. Johnson, who gave me their love and support throughout my studies, and especially to my wife, Lily M. Johnson, whose unmitigated love and encouragement kept me going.

To my Father, Charles D. Johnson, career military man of the U.S. Air Force, veteran of the Korean and Vietnam wars, my role model, my mentor, I know you can see me from up there.

To my Mother, Florence M.J. Johnson, her firm hand and nurturing love made me who I am.

To my friend and colleague, Robert A. Piirainen. We got ourselves into what?

To Professors Luqi, Valdis Berzins, and Mantak Shing, for seeing in me the potential to succeed, and for their uncompromising encouragement and support.

## Robert A. Piirainen

To my parents Wayne H. Piirainen, my father, who was always proud to have served in the United States Navy during World War Two and my mother Lena M. Piirainen who was always there. Both taught and lived the virtues of truth and courage, SISU.

To my friend, collaborator, and colleague, Craig Johnson who I could not have done this without.

To my supervisors, colleagues, and friends Benedict DeMarco, Dr. Chong Liao, Ejaz Qureshi, and Abdul Siddiqui whose support and collaboration was essential in this as it has been in all that we do.

To my fellow students and the faculty at NPS who all inspired me and from whom I learned much.

To Professors Luqi, Valdis Berzins, and Mantak Shing who by their inspiration, guidance, and example were able to teach an old crusty engineer much.

# I. INTRODUCTION

## A. PURPOSE

The purpose of this Thesis is to assess the practical use of the Nogueira formal model for risk assessment on real-time embedded software development projects. The use of formal models for assessing risk in software projects is an emerging methodology. In the case of the Nogueira model for risk assessment, application to real-world software development projects has been extremely limited. In that light, this Thesis presents the model's post mortem application to real-world software development projects. An analysis of how well the model tracks to the actual software's development is documented, compared and contrasted against simulated data from the Nogueira dissertation.

## B. PREPARATION

We found that developing project selection criteria, and a data collection plan, essential in selecting viable projects for our Thesis to meet the timelines and resources available. These criteria prevented us from pursuing false or untenable paths. Further, we found that having a strong Integrated Product Team (IPT) relationship with the developer essential to collecting the necessary data. If this does not exist, significant time and effort must be planned for to establish the necessary relationships. In future work, tailored selection criteria and data collection plans could be developed for new timelines and resources and for applying this method to projects at their beginning instead of post mortem. In addition, we could investigate developing plans for projects developed without Object-Oriented techniques and CASE Tools as well.

## C. BACKGROUND

In the software evolution domain, risk assessment has not been addressed as part of previous models. In the various enhancements and extensions, the graph model does not include risk assessment steps; hence, risk management remains a human-dependent activity that requires expertise.[2] Risk is defined as the product of a future outcome times the probability of an occurrence of such an outcome. The outcome could be negative, a loss (this is the general approach that all previous research has applied), but could also be positive leading to a gain.[3]

As the range and complexity of computer applications have grown, the cost of software development has become the major expense of computer-based systems (Boehm, 1981), (Karolak, 1996). Research shows that in private industry as well as in government environments, schedule and cost overruns are tragically common (Luqi, 1989), (Jones, 1994), (Boehm, 1981). Developing software is still a high-risk activity. Despite the advances in technology and CASE tools, little progress has been done to improve managing software development projects (Hall, 1997). The acquisition and development communities, both governmental and industrial, lack systematic ways of identifying, communicating and resolving technical uncertainty (SEI, 1996). Research shows that 45 percent of all the causes for delayed software deliveries are related to organizational issues (vanGenuchten, 1991).[4]

---

[2] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 5

[3] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 4

[4] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 20

Current experience suggests that building and integrating software by mechanically processable formal models leads to cheaper, faster and more reliable products (Luqi, 1997). Software development processes, such as the Hypergraph model for software evolution (Luqi, 1997) and the Spiral model (Boehm, 1988) have improved the state of the art. However, they share a common weakness: risk assessment.[5]

In this Thesis we focus on the Nogueira model, as it is a formal model for risk assessment for use in software projects, and its creation was based on addressing the risk assessment issue. Presently, the Nogueira model provides a theoretical basis for automatically assessing and identifying risks early in a project's development. It is this aspect that we are trying to validate, post mortem, and is the focus of our Thesis.

## D. RESEARCH QUESTION

Can the Nogueira model for formal risk assessment be applied to real-world projects with a degree of assurance that it can predict risk that correlates closely to real data?

## E. SCOPE

The scope of this Thesis encompasses the collection and gathering of software project data from two real world, real-time, embedded software development efforts. It then feeds that data into the model for analysis and prediction of risk as related to the probability of successful completion given the original schedule constraints. As this is a post mortem of existing projects, the actual project completion data are known. It was expected that the comparison of the actual known data would have a high correlation to

---

[5] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 1

the Nogueira model predictions and would contribute to the Nogueira model's refinement, calibration, and validation for its further use in real world applications.

## F. ORGANIZATION

This Thesis is organized into the following chapters:

- Chapter I presents a brief but concise introduction to our Thesis composition and organization. It contains a succinct description of the purpose, background, and scope of this Thesis, as well as presenting our research question.

- Chapter II presents an overview of previous research focused on software risk assessment on which this Thesis is based. We have leverage the prior work of Dr. Juan Carlos Nogueira, inventor of the Nogueira Formal Model for Risk Assessment.

- Chapter III introduces our detailed selection criteria and the reasoning behind the selection of our two candidate software projects, as well as, the application of the Nogueira formal model for risk assessment to the two selected software projects.

- Chapter IV presents the actual Nogueira formal model computations and calculations. It presents the ideals of Complexity (CX), in the form of Large Granularity Complexity, Requirements Volatility (RV) as, calculated by the Change Rate (CR), related to Birth and Death rates of requirements, as well as, their use to feed the Nogueira model for prediction of completion probabilities.

- Chapter V presents our data collection methods and assumptions. It demonstrates our use of the Nogueira formal model as fed by the input metrics of Requirements Volatility (RV), Complexity (CX), in the form of Large Gain Complexity. (LGC) and the Efficiency Factor (EF), detailed calculations, using the actual data

4

collected from the two selected software projects, are given here. In addition, a detailed comparison and contrast of the two selected software project's Bi-dimensional plots are provided.

- Chapter VI presents a comparison of our two project's calculated LGCs versus the Nogueira LGC equation. It provides our explanation as to any differences or similarities seen when the comparisons are made.

- Chapter VII presents the thought process and logic behind our determining the Nogueira Efficiency Factor (EF) metric assumption. We made an assumption because it was necessary to allow us to continue our Thesis research. Details as to why the assumption was made are given here.

- Chapter VIII presents the actual analysis of the Nogueira formal model's predicted probability of completion when fed in actual metrics calculated by actual project data. The model's estimated probability of completion is compared to the actual data collected from our two software projects, and our explanation as to any differences or similarities seen when the comparisons are made is given here.

- Chapter IX presents a comparison of our two project's actual code sizes versus the Nogueira estimations. It provides our explanation as to any differences or similarities seen when the comparisons are made.

- Chapter X presents conclusions, a summary of the contribution of this effort, lessons learned from this effort, and a summary of the findings of this research, answers to the research question, and recommendations for further research and study.

THIS PAGE LEFT INTENTIONALLY BLANK

## II.    PREVIOUS WORK

This Thesis applies previous research[6] on software project risk assessment, namely predicting the success of the project.  The only ways to evaluate the degree of success of a project are (a) to compare planned and actual schedule, (b) to compare planned and actual costs; and (c) to compare planned and actual product characteristics. Software reliability, an emergent branch of software engineering, has addressed this last part. However, the first two issues have not yet been emphatically addressed.

For many years research has greatly increased our knowledge of software projects. Among such software laws, it is known that:

- Manpower and time are not interchangeable (Brooks, 1974).

- Human productivity rates are highly variable, and function and size are highly correlated with errors and duration of the project (Putnam, 1980).

- The majority and most costly errors are introduced during the requirements phase (Boehm 1981).

- Life-cycle manpower patterns follow tailed probability curves (Norden, 1963), (Putnam, 1980, 1992, 1996, 1997), (Boehm, 1981).

- Standards, good practices, guidelines, and heuristics improve the development process (Humphrey, 1989).

At present, there are Computer Aided Software Engineering (CASE) tools that can improve productivity.  Also, Macro models can estimate, with different degrees of success, the effort and duration of software projects (Albrecht, 1979), (Boehm, 1981,

---

[6] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 1

2000), (Putnam, 1997). What is not available is a model of the internal phenomenology of the software life cycle. Without the knowledge of such a model, scientific risk assessment is almost impossible. The software process is a set of activities with dependency relationships that occur over a certain period of time. From this point of view software development projects do not differ from any other type of project. At the beginning of such a process, a great deal of uncertainty exists. This uncertainty can be reduced through effort, which can be expressed as time and cost. As time goes by, the level of uncertainty usually decreases because more information becomes available.

Unfortunately, the main resources (time and budget) are interrelated and must be traded off. So project managers, as decision-makers, must choose between making early decisions with a great deal of uncertainty, or postponing decisions by trading time for information. The concept of early measure is emphasized because recognizing the risks in the early phases of software development increases the probability of success, and therefore, improves the competitive advantage. The focus of this research is on automatically collectable measures since risk identification should be as objective as possible and not impose any substantial additional workload.

## III. APPLICATION OF THE FORMAL MODEL

After studying the Nogueira Dissertation, and before considering any candidate projects for this Thesis, we consulted with Dr. Juan Carlos Nogueira. As a result, we established a set of desired criteria to help in determining which were the most viable projects for application of the model.

### A. PROJECT SELECTION CRITERIA

The following selection criteria were used in selecting the projects for our study:

- The projects were recent DoD development projects utilizing the Software Engineering Institute, Capability Maturity Model, (CMM)[7] level 2 processes or better which had the necessary metrics and data available (figure 1.)
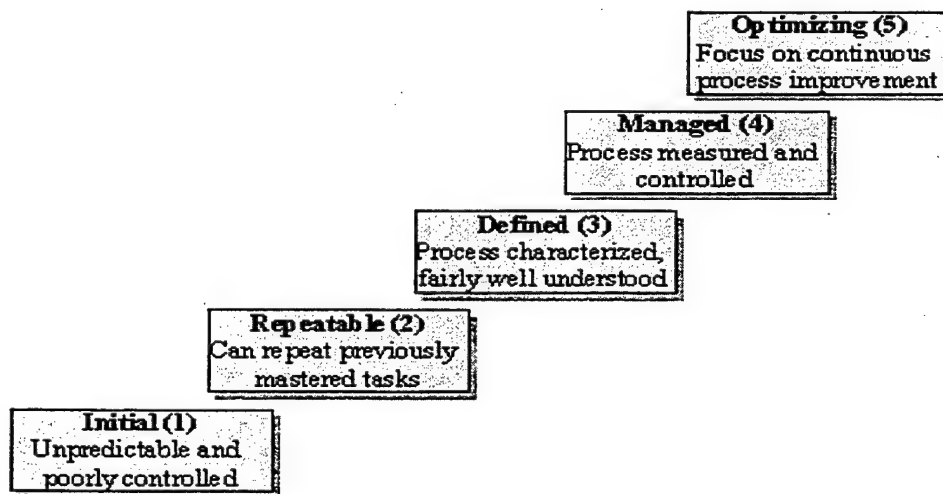


Figure 1 Five Levels of the Capability Maturity Model (CMM)

- The projects had worked through the lifecycle phases of an evolutionary development, in this case an evolutionary spiral model[8].

---

[7] Carnegie Mellon University, Software Engineering Institute, The Capability Maturity Model (Addison Wesley:1994) 32-35.
[8] Department of the Airforce STSC - Guidelines for successful acquisition and management of software intensive systems, version 2.0, dated June 1999, Chapter 3

- The project used Object-Oriented Methodology (OOM)

- The project contained multiple Computer Software Configuration Items (CSCIs)

- The project was Coded in Ada

- The software was real-time embedded

- The project used a Computer Aided Software Engineering, (CASE) tool.

## B.  THE CANDIDATE PROJECT SELECTION PROCESS

Three real projects were considered as possible candidates for our research, with the specifics for evaluation of each one listed below:

- The first project used an Evolutionary Spiral lifecycle model. It used Object-Oriented methodology. It was composed of five Computer Software Configuration Items. It used Ada. It was real-time embedded, it used Rational Rose as a Computer Aided Software Engineering tool, and the developer was operating at SEI level 3. In addition, software metrics from three builds over a period of three years had been kept. This project met all of our ideal project selection criteria; we selected this as our first candidate project.

- The second project originally used an incremental build lifecycle model and not an evolutionary model. It originally used Functional Decomposition methodology. It was composed of six Computer Software Configuration Items. It used Ada and assembler. It was real-time embedded. It did use upper CASE tools, like Requirements Traceability Matrix (RTM), it did not use lower CASE tools such as Rational Rose. The development effort initially

was performed in an Ad-hoc manner with little software process involved and had experienced extreme volatility and poor metrics early in its development. However, due to a major restructure and overhaul of the project, and a shift of focus to institutionalizing software processes, (SEI CMM level 3 certification), the project migrated to Ada, and began using a modified Incremental Build lifecycle model. In addition, suitable software metrics from two recent builds were available. Although this project did not meet all of our ideal project selection criteria, i.e., it did not use a lower CASE tool and had limited available metrics, we selected this as our second candidate project because we felt the data was sufficient to analyze and draw conclusions.

- The third project used an Incremental Build model. It used functional decomposition. It was composed of five Computer Software Configuration Items. It used Ada. It was real-time embedded, and used only an upper CASE tool, RTM; it did not use A lower CASE tool like Rational Rose. The project had just completed analysis and design, and was beginning code development. It had not completed a first build yet, so consequently metrics were insufficient for our use. Most importantly, actual complete times were not yet known, and would not be within the schedule constraints of the current research. Consequently, we deemed this project unsuitable for our study until it progresses further in its development.

## C.     ADDITIONAL FACTS OF THE TWO PROJECTS SELECTED

### 1.     Candidate Project One

The first project selected never completed as it was defunded due to external Department of Defense considerations.  Built In Test (BIT) Diagnostics requirements were deferred at the beginning of the second build until the third build and then to a forth build which was never built due to the defunding.  Because of this, we believed the first and second builds were corrupted for the purposes of calculating accurate durations. Therefore, even though Requirements Volatility (RV), Birth Rates (BR), and Death Rates (DR) were calculated for the three builds, only build three's data was used to validate the Nogueira model against our project.  Nevertheless, we did use all of the three builds data to generate a bi-dimensional plot for purposes of requirements stability analysis and correlation to actual project events.  An analysis of the bi-dimensional plot data is given later in section V.A.1 of this Thesis.

### 2.     Candidate Project Two

The second project selected, and the organization developing it, had matured over a six-year period from a Software Engineering Institute (SEI) Capability Maturity Model (CMM) Level 1 to a level 3. Consequently, they had not collected the necessary data required for this study earlier in the project.  Data was available for the period March through October 2000, which encompassed the late stages of one build and the development of a second build. Core functionality on three Computer Software Configuration Items (CSCIs): Fire Control (FC), System Manager Software (SMS), and Operating Environment (OE) had been developed and validated. The software builds during this period involved addition of functionality to the Command and

12

Communications (C2), Vehicle Diagnostics Management System (VDMS), and Soldier Machine Interface, (SMI) CSCIs. Also, Problem Change Requests (PCRs) were corrected in this time frame. In addition, this project had gone through extensive testing. and user trials at government test sites; processes and requirements were stable and consequently requirement volatility was low. This data is further described in a bi-dimensional plot given in section V.A.1 of this Thesis.

THIS PAGE LEFT INTENTIONALLY BLANK

# IV. THE CALCULATION OF THE NOGUEIRA MODEL PARAMETERS

## A. THE PHENOMENA

When using the third iteration of the Nogueira model, we noted strange phenomena occurred in the model. When RV exceeded 40 percent, the Beta element of the algorithm went negative reversing the probability prediction. To verify that this was truly occurring, we continued to input additional higher RV percentages. As the higher RV percentages were input, the model began predicting shorter completion times, and higher probabilities of success, which project experience and logic would dispute. As such, we decided to consult Professor Berzins on our discovery. We were able to repeat the phenomena for Professor Berzins, and this resulted in the refinement of the third model's algorithm thus creating the fourth iteration of the Nogueira model.

In this Thesis we're assessing the updated fourth model. For the purposes of a tool, we originally programmed an MS Excel spreadsheet with the fourth algorithm of the Nogueira model to allow us to exercise "what if" scenarios. However, our tool was limited in that it did not generate a graph depicting the probability curve. Another student at NPS, CPT Michael Murrah, took our original spreadsheet model and extended it. The new tool combines the effects of all four models graphically and by data, (figure 2.) This tool is available from the Naval Postgraduate School.

**All Models**

Chart: P(t) vs Time (0–80), with lines for Model 1, Model 2, Model 3, Model 4, and 50.0.

| Model 4 | |
|---|---|
| **Efficiency** | |
| Direct Labor % | 67.00% |
| Idle % | 33.00% |
| EF | 2.030 |
| **Req. Volatility** | |
| Birth Rate % | 2 |
| Death Rate % | 6.8 |
| RV | 8.800 |
| **Complexity** | |
| Operators | 500 |
| Data Streams | 177 |
| Types | 200 |
| LGC | 877 |

| Model 1 | | Model 2 | | Model 3 | | Model 4 | |
|---|---|---|---|---|---|---|---|
| Alpha | 1.95 | Alpha | 1.95 | Alpha | 1.95 | Alpha | 1.95 |
| Beta | 6.57562964 | Beta | 6.129824241 | Beta | 7.956519821 | Beta | 8.499165678 |
| Gamma | 36.16596302 | Gamma | 36.16596302 | Gamma | 41.33252917 | Gamma | 48.53023602 |
| P(t) | 98.59% | P(t) | 99.25% | P(t) | 69.32% | P(t) | 3.21% |
| t (days) | 50.0 | t (days) | 50.0 | t (days) | 50.0 | t (days) | 50.0 |
| months | 2 | months | 2 | months | 2 | months | 2 |

**Model 1\***

Used when RV is small.

\*CX >= 600 LGC
\*RV <= 67%

**Model 2\***

Considers the three factors, but neglects combined effects of (EF & RV)

**Model 3\***

Considers the three factors and combines their effects

**Model 4\*\***

Model used for any range of complexity and requireme(nts) volitility, and considers all factors, with their combined effects.

\*\*A 0 LGC will take 0 days
\*\*Alpha, Beta, Gamma > 0
\*\*If RV increases then P(t) de(creases)
\*\*If CX increases then P(t) de(creases)
\*\*If EF increases then p(t) inc(reases)

Figure 2    Nogueira Formal Model Tool (Ms Excel)

## B.    THE ESTIMATING MODEL PARAMETERS CALCULATIONS

We determined the two-selected project's predicted completion times by calculating two of the three input metrics from the Nogueira model: Complexity (CX) and Requirements Volatility (RV). The third input metric, Efficiency (EF) was assumed, and an explanation of why is given in section VII of this Thesis.

We collected the following data for use in the input metrics calculations. Independent data collection from the developer was realized via already established

developer and Government relationships. In essence, we had full access to all of the project's data.

THIS PAGE LEFT INTENTIONALLY BLANK

## V.    THE REQUIREMENTS VOLATILITY (RV) METRIC CALCULATION

We calculated the Requirements Volatility (RV) input metric using the Nogueira model method[9]. The RV input metric was calculated using data collected on the projects. The specifics of each project's RV calculations are as follows:

## A.    BIRTH RATE (BR), DEATH RATE (DR), AND REQUIREMENTS VOLATILITY (RV), ANALYSIS:

### 1.    First Project

For the first project, the RV input metric was calculated using data collected on the three separate builds. Below are the specifics of the builds:

| 3 Builds | Definitions |
|---|---|
| Build 1, 10/23/98 – 12/18/98 | RV = BR % + DR % |
| Build 2, 01/01/99 – 02/26/99 | Birth Rate, BR |
| Build 3, 02/26/99 – 01/28/00 | Death Rate, DR |
| | BR = (Added Rqmts + Changed Rqmts) / Total Rqmts x 100 % |
| | DR = (Deleted Rqmts + Changed Rqmts) / Total Rqmts x 100% |

First project's requirements volatility data by month

| Month Year | Oct 98 | Nov 98 | Jan 99 | Feb 99 | Mar 99 | Apr 99 | May 99 | Jun 99 |
|---|---|---|---|---|---|---|---|---|
| RV% | 81.3% | 1.5% | 75% | 7.6% | 1.2% | 0% | 3.1% | 4.8% |
| BR% | 22.8% | 0.5% | 35% | 3.7% | 0.8% | 0% | 1.3% | 3.0% |
| DR% | 58.5% | 1.0% | 40% | 3.9% | 0.4% | 0% | 1.8% | 1.8% |

| Month Year | Jul 99 | Aug 99 | Sep 99 | Oct 99 | Nov 99 | Dec 99 | Jan 00 |
|---|---|---|---|---|---|---|---|
| RV% | 0% | 0% | 53.3% | 3.03% | 29.8% | 5.45% | 17.7% |
| BR% | 0% | 0% | 19.7% | 1.56% | 15.0% | 2.93% | 8.0% |
| DR% | 0% | 0% | 33.6% | 1.47% | 14.8% | 2.52% | 9.7% |

First project's requirements volatility data by software build

| Build 1 (O98 – N98) | Build 2 (J99-F99) | Build 3 (M99-J00) | Project |
|---|---|---|---|
| RV = 33.6 % | RV = 41 % | RV = 9.6 % | RV = 16.8% |
| BR = 12.5 % | BR = 19.3 % | BR = 4.5% | BR = 7.5% |
| DR = 21.1 % | DR = 21.7 % | DR = 5.1% | DR = 9.3% |

Please note that we decided to use only Build 3 data for our model calculations and comparisons due to the fact that builds 1 and 2 were "manipulated." In this case, requirements were deferred to build 3 and eventually to build 4, which was never built because of deprogramming. Schedule and Cost requirements (over Performance) were met by deferring BIT requirements, stabilizing the core product, then introducing BIT requirements. This last step was not accomplished because of the previously mentioned defunding.

A bi-dimensional plot of the three builds was made to note trends at a macro level (figure 3.) Builds 1 and 2 fell in the volatile region while build 3 was in the stable region.

[9] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 117
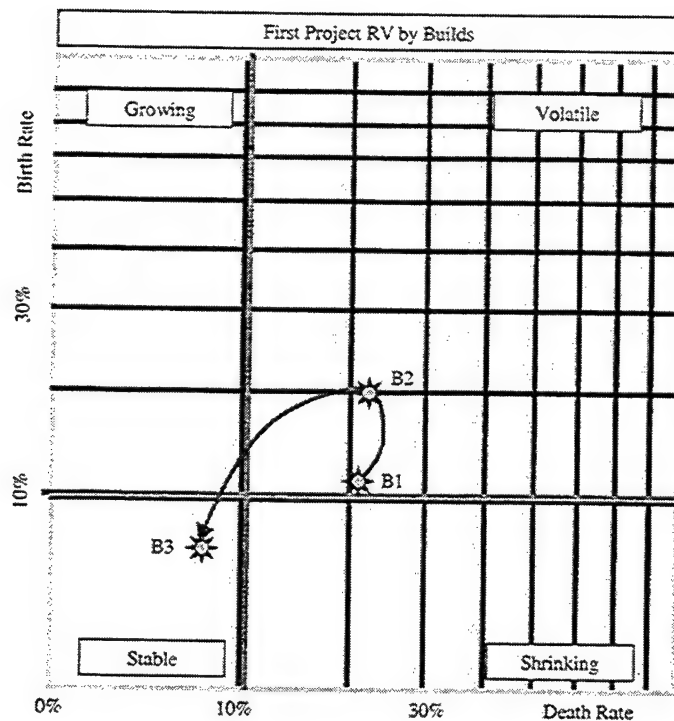
Figure 3 First Project RV by Builds

This correlated with the removal of the BIT Diagnostics requirements at the beginning of build 2. Deferring the BIT Diagnostics until a later build after the core product became stable was arguably an astute move; however, it would have been better to have this in the planning up front.

So, it appears trying to develop your diagnostics software before your core product is stable is not prudent, better to defer until the core product is stable. We believe the bi-dimensional plot is a useful tool for comprehending this. If this project had this tool they may have been able to come to grips with this sooner. Future work with the bi-dimensional plot could help demonstrate and make generally known the impacts of developing diagnostics before the core product is stable.

A bi-dimensional plot of build three by month (figure 4) was made to note trends at a micro level. BR and DR data for the eleven months of this build, Mar 1999 through January 2000 were plotted.
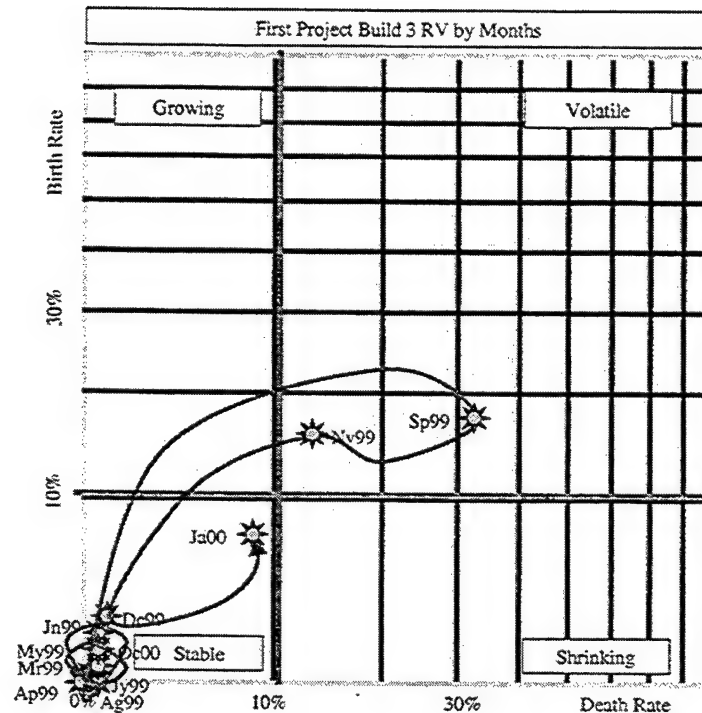


Figure 4  First Project - RV Plot Build 3 By Month

This Bi-dimensional plot shows build three's volatility trending into the stable region, two months September 1999 and November 1999 went into the volatile region with the last month of this build January 2000 returning to the stable region. The requirements volatility cycled from stable to unstable twice with the overall trend to the stable region. It appears that deferring the diagnostics had a stabilizing effect. This trend, at the micro level, is behaving like a dampened sinusoidal (figure 5.)
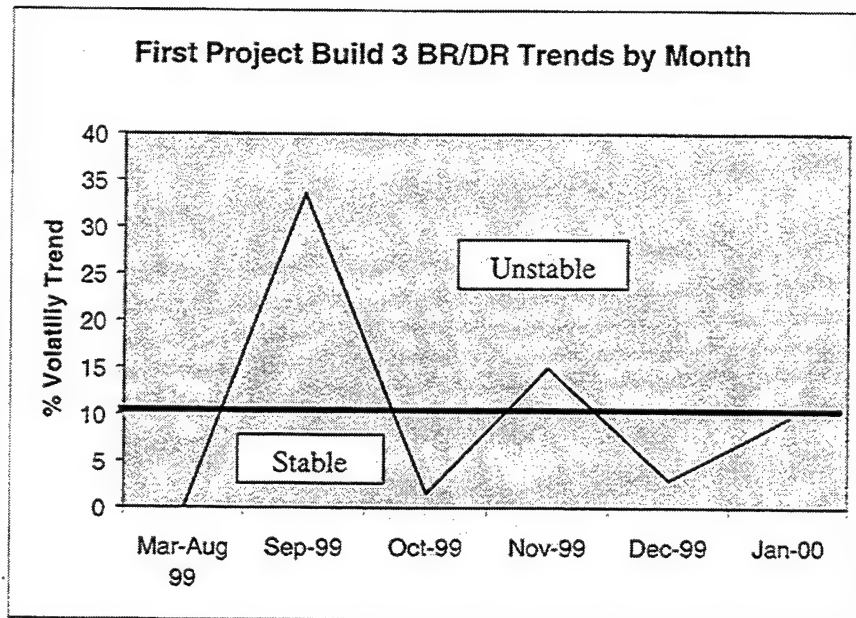
Figure 5 First Project Build 3 BR/DR Trends by Month

This would have been an interesting phenomenon to observe to confirm if the trend continued and that requirements indeed stabilized and stayed in the stable region of the Bi-dimensional plot. We believe this would have been the case. We believe these cycles were process driven. However, this does highlight the importance of not reacting to limited data points at a micro level and that people evaluating project metrics should have a working knowledge of projects and their processes, otherwise wrong conclusions can be reached and counterproductive actions could be taken. We believe actions by managers here would have likely worsened the situation. The Bi-dimensional plot has proven to be a useful tool for identifying trends and assessing relative volatility versus looking at data in tables.

## 2. Second Project

For the second project, the RV input metric was calculated using monthly data
collected over an eight-month period.

| Eight Months | Definitions |
|---|---|
| Build 1 Mar – May 2000<br>Build 2 Jun - Oct 2000 | RV = BR % + DR %<br>Birth Rate, BR<br>Death Rate, DR<br>BR = (Added Rqmts + Changed Rqmts) / Total Rqmts x 100 %<br>DR = (Deleted Rqmts + Changed Rqmts) / Total Rqmts x 100% |

Second project's Requirements Volatility Data by Month

| Month<br>Year | Mar 00 | Apr 00 | May 00 | Jun 00 | Jul 00 | Aug 00 | Sep 00 | Oct 00 |
|---|---|---|---|---|---|---|---|---|
| RV | 0% | 0% | 0% | 3.24% | 4.47% | 5.58% | 5.65% | 9.22% |
| BR | 0% | 0% | 0% | 1.16% | 1.90% | 2.33% | 2.41% | 5.14% |
| DR | 0% | 0% | 0% | 2.08% | 2.57% | 3.25% | 3.24% | 4.08% |

Second project's Requirements Volatility Data By Software Build

| Build 1 (March – May) | Build 2 (June – Oct) | Project |
|---|---|---|
| RV= 0% | RV= 3.24 % - 9.22% | 5.63% |
| BR= 0% | BR= 1.16% - 5.14% | 2.59% |
| DR= 0% | DR = 2.08 – 4.08% | 3.04% |

A bi-dimensional plot of the two builds was made to note trends at a macro level

(figure 6.)
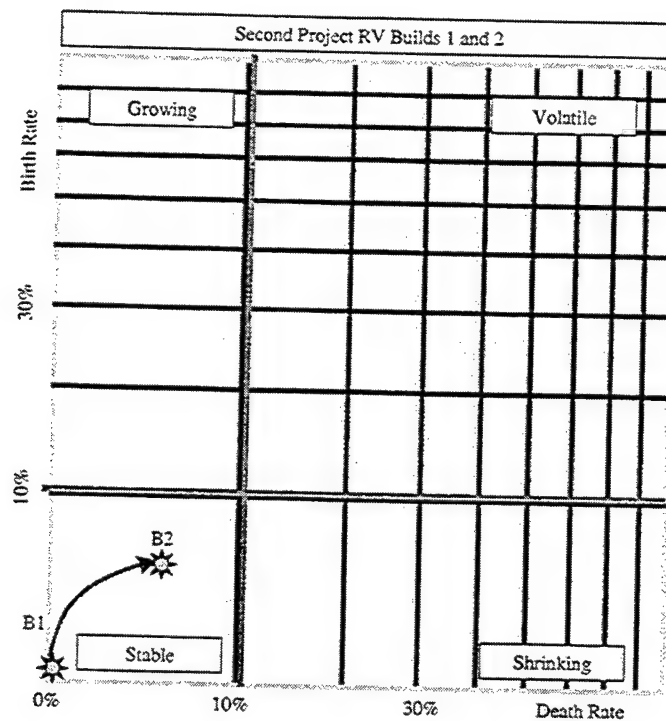
Figure 6  Second Project's Bi-Dimensional Plot of the 2 Builds

The Requirements Volatility Data from this eight-month timeframe captures the end of a software incremental build. March, April, and May, which showed RVs of 0 %, represents a timeframe when a build was nearing completion and changes to requirements had already been identified and corrected.

When the next build began in the June through October timeframe, the renewed activity is seen in the RV calculations, which shows the new functionality requirements being refined and corrected. A bi-dimensional plot of build two by month was made to note trends at a micro level (figure 7.) BR and DR data for the five months of this build, June 2000 through October 2000 were plotted.
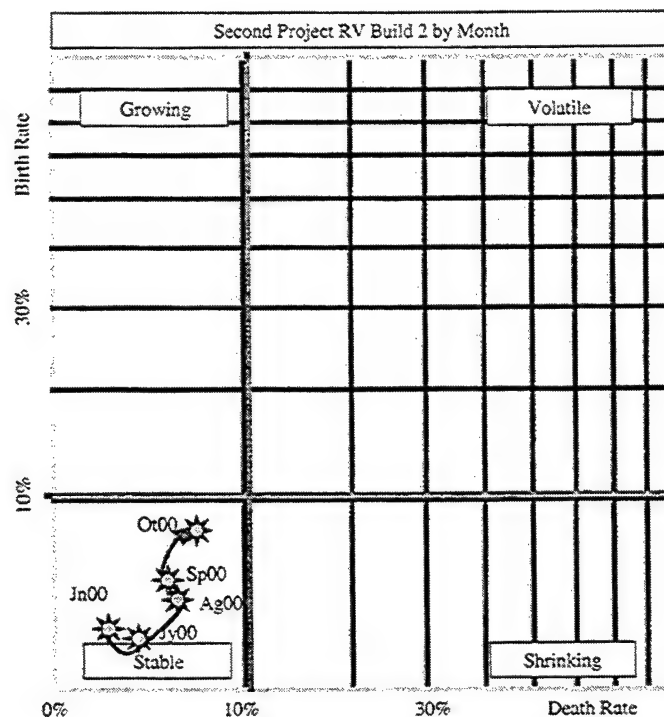
Figure 7 Second Project's Build 2 By Month

All BR, DR plots fall in the stable region of the bi-dimensional plot with a slight

trend up as the build progressed. This slight trend up is attributed to the introduction of

new Command and Control (C2), and Soldier Machine Interface (SMI) functionality, as

well as, the fact that vehicle diagnostics were also being matured. Further, we believe

this slight trend up in the requirements volatility, as the build progressed, reflects a CMM

level 3 organization refining and documenting the requirements for the new functionality.

In situations such as this, it is prudent to check the nature of the changes to ascertain if

they are major or minor changes. In this case, most changes were minor which

collaborated the opinion that requirements were stable. Moreover, it is worth noting that

this software build was Configuration Management (CM) released, Formal Qualification

Tested (FQT), government stress tested, and successfully underwent user operational

testing. However, additional bi-dimensional plots from other projects and releases would be helpful in developing the use of, and validity of bi-dimensional plots.

We find the bi-dimensional plot a useful tool to analyze RV data with project activity and validate expert opinion. Of interest and note is that the search and analysis of RV data was done in the same development organization that had achieved a SEI CMM level 3 rating. Data was not universally kept across the three projects, the first project had kept RV data from the outset, and this may have been due in part because it was started after the organization had achieved a CMM Level 2 rating and used a lower CASE tool. The other two projects had started keeping the data relatively late, March 2000.

The developers pursuit of CMM level 4 is having a positive effect on the collection of RV data; from this we surmise that application of the Nogueira method is at the level of CMM level 4. The conclusion we are drawing from this is that even with an organization that is a CMM level 3, and is committed to achieving CMM level 4, a Project Management Office cannot expect to get RV analysis soon enough to proactively manage a project. They still must rely on experienced software managers who can assess project status real-time based on expert opinion. At present, this RV data is useful to confirm that our management and process is on track but not timely enough to be useful as a management tool. Also, in untrained hands this RV data could lead to erroneous conclusions. W. Edwards Deming pointed out, variance occurs, and if untrained managers cannot tell the difference between variance and trends they can wind up chasing variance and severely hampering a project.[10]

---

[10] W. Edward Deming, Out of the Crisis (MIT Press Nov 1989) 309

Interest in keeping the data has been renewed since a CMM level 4 audit pointed out process deficiencies and their desire to achieve a CMM level 4 should tighten this up. The conclusion we are drawing from this is that even with an organization that is a CMM level 3, and is committed to achieving CMM level 4, a Project Management Office cannot expect to get RV analysis soon enough to proactively manage a project. They still must rely on experienced software managers who can assess project status real-time based on expert opinion. At present, this RV data is useful to confirm that our management and process is on track but not timely enough to be useful as a management tool. Also, in untrained hands this RV data could lead to erroneous conclusions. W. Edwards Deming pointed out, variance occurs, and if untrained managers cannot tell the difference between variance and trends they can wind up chasing variance and severely hampering a project.[11]

A future research opportunity would be to gather a statistically significant RV database, from many projects, and establish guidelines on assessing RV plots, variances, and trends. We believe it is unrealistic and unwise to expect to, or endeavor to, eliminate the expert opinion of experienced software practitioners. Additionally, we believe the wise and realistic goal is to develop tools that will help them to perform their jobs more effectively. Further we believe the RV, BR, DR, and bi-dimensional plot has the potential to do this but guidelines based on statistically significant project data should be developed before this tool is released, otherwise it is likely to cause more havoc then help.

---

[11] W. Edward Deming, Out of the Crisis (MIT Press Nov 1989) 309

## B.   REQUIREMENTS STABILITY

As with the Bi-dimensional plot, additional charts are shown in this section to demonstrate the specifics of the volatility of requirements throughout each of the selected project's software development lifecycles.

### 1.   First Project

As you look at the chart (figure 8) you will observe that there are fairly high volatility rates starting at 33.6% for build one and increasing to 41% for build two, then dramatically dropping in build three to 9.6%.   This correlates with the previous explanation of the bi-dimensional plot, as research indicated that in the first year and a half of the project's effort, intercommunication between Systems Engineering and Software Engineering was nearly nonexistent.   In essence, it created a situation where system requirements were being allocated down to software without input from Software Engineering.   This resulted in copious re-work at the software requirements level as the system level requirements were constantly being changed.   Management finally recognized that this was a significant problem when the project started to slip schedule by approximately a year, and exceeded budget by one million dollars.

**First Project's RV by Builds**
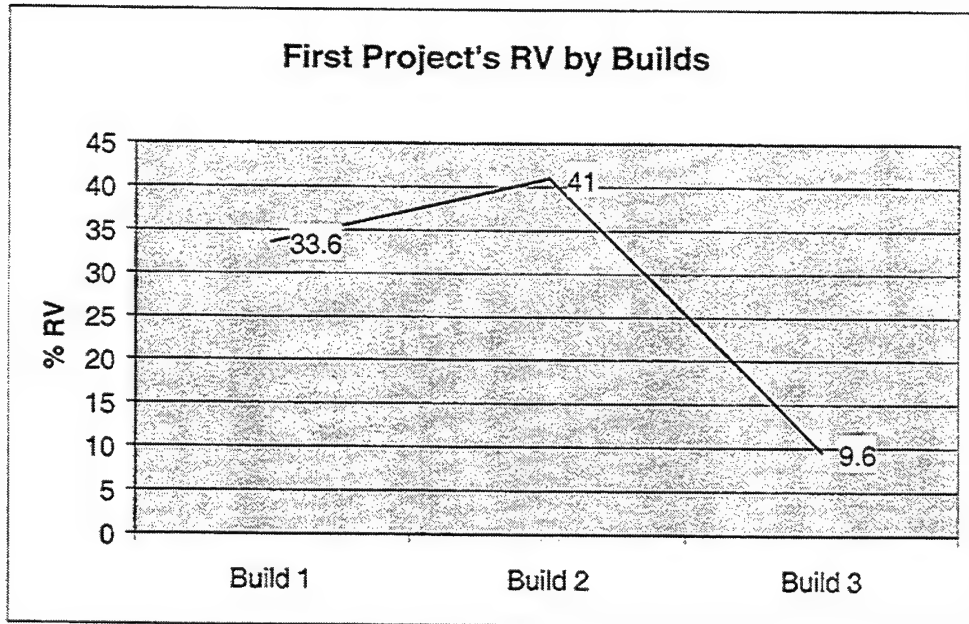
Figure 8  First Project's Rv Plot by Builds

At this point management took radical action to correct the intercommunications problem by establishing an Integrated Product Team (IPT) enabling the involvement of System Engineering with Software Engineering. An Engineering Review Board (ERB) was established where all changes had to be formally boarded and agreed to before a requirement could be added or changed. The ERB was co-chaired by the lead Systems Engineer and the lead Software Engineer. This dramatically reduced the rate at which requirements were changed, leading to a fundamental stabilization of the system and hence software requirements. In addition to this, as stated before, significant functionality - BIT Diagnostics, was deferred in order to meet schedule and cost limitations. An additional chart is included (Figure 9) illustrates the stabilization trend occurring from September 1999 to January 2000.

**First Project RV By Month**
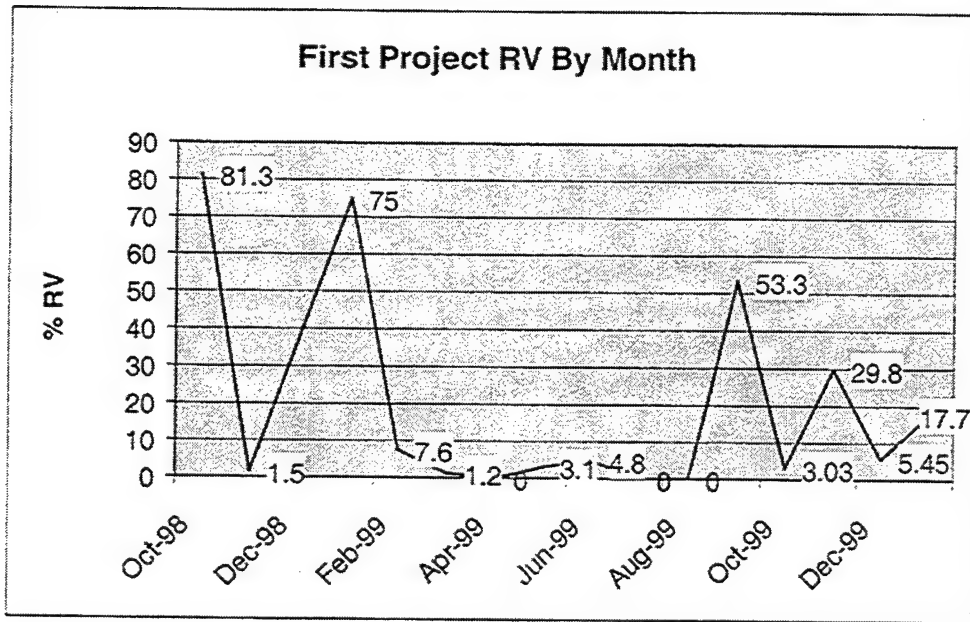
Figure 9 First Project RV by Month

## 2. Second Project

The RV plot shows 0% for a software release that was at the end of its development cycle (figure 10.) This indicates that the new functionality requirements had been refined, corrected, and completely stabilized. The higher RV data is at the beginning of a new software release where new functionality (C2, SMI and vehicle diagnostics) requirements are being refined in design. The second chart (figure 11) shows RV within build 2, but broken down by month, to explain why the trend up occurred. As discussed in the bi-dimensional plot, the BR and DR plotted into the stable region. Traditional volatility data only indicates trends over time; it gives no guidelines on how to objectively assess project health. As with the bi-dimensional plot data, volatility data in the hands of managers who do not understand Total Quality Management Theory[12] can result in chasing variance and a waste of resources.

---

[12] W. Edward Deming, Out of the Crisis (MIT Press Nov 1989) 309
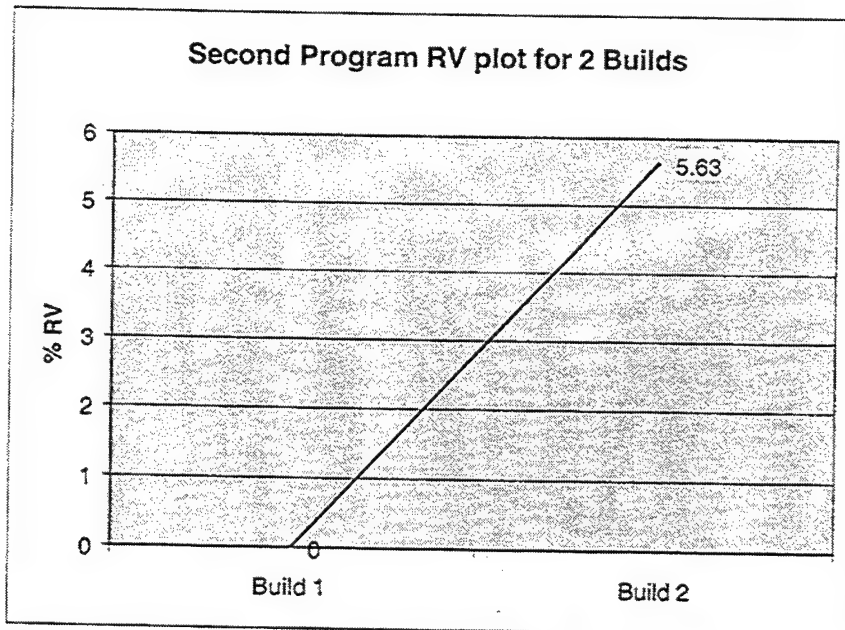
Figure 10        Second Project RV Plot Builds 1 and 2
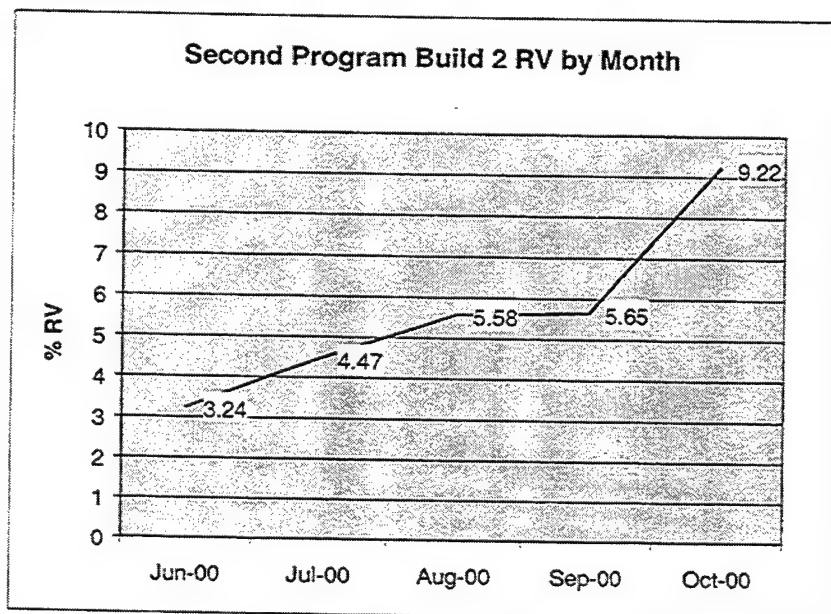


Figure 11        Second Project Build 2 RV by Month

## C. THE CX METRIC CALCULATION

### 1. The Data Collection Method

Since this was a post mortem application of the Nogueira model, we had to devise a means to collect the required "Operators, Data Streams, and Abstract Data Types" data for input to the Large Granularity Complexity (LGC) metric. As we did not have anything like Prototype System Description Language (PSDL)[13] or a Directed Hypergraph, we could not count Operators, Edges, and Data Streams, as you could utilizing a tool like the Computer Aided Prototyping System (CAPS.)[14] Therefore, we came up with the following method to count certain entities in the Ada source code that represented Operator, Data streams and Types. In essence, we took a representative sample of our project's Ada source code and performed an analysis of it. We determined that words in the source code like PROCEDURE, TASK, and PACKAGE, represented "Operator" (O) entities, and TYPE represented "Type" (T) entities and MESSAGE represented "Data stream" (D) entities. We further surmised that a count of PROCEDURE, TASK, PACKAGE, TYPE and MESSAGE for our particular domain would get us an accurate enough count for input to the Nogueira model. Since our host system was UNIX based, we decided to use a UNIX "grep" command to extract the data. We used the form:

*grep –i 'string' \*.ada | wc –w*

Where *'string'* is replaced by the word to be looked for i.e. PROCEDURE, TASK, PACKAGE, TYPE, and MESSAGE etc., and *wc* is the word count by word *–w*.

---

[13] Luqi et al., <u>Computer Aided Prototyping System (CAPS) for Heterogeneous Systems Development and Integration</u> (Naval Postgraduate School, 2001) 1-13

[14] Luqi et al., <u>Computer Aided Prototyping System (CAPS) for Heterogeneous Systems Development and Integration</u> (Naval Postgraduate School, 2001) 1-13

We were allowed access to the developer's Ada source code library on the UNIX server with read only permissions. We accessed the directories by Computer Software Configuration Item (CSCI) name. For each CSCI, we ran the grep command for PROCEDURE, TASK, PACKAGE, TYPE, and MESSAGE on all associated Ada files. We then totaled all the counts and obtained the Operators (O), Data Streams (D), and the Abstract Data Types (T). The O input consisted of the combined counts of PROCEDURE, TASK, and PACKAGE. The D input consisted of the total counts of MESSAGE. And the T input consisted of the total counts of TYPE.

## D. FIRST PROJECT

CX- (Complexity in LGC=O+D+T)   LGC – Large Granularity Complexity

For the total, all build 3 Computer Software Configuration Items (CSCIs), were used to determined the following counts:

| **O** = Operators | **D** = Data Streams | **T** = Abstract data types | **LGC** =**O+D+T** |
|---|---|---|---|
| **841** total operators | **1468** total data streams | **483** total Data Types | 2792 |

## E. SECOND PROJECT

CX- (Complexity in LGC=O+D+T)  LGC – Large Granularity Complexity

For the total, all five Computer Software Configuration Items (CSCIs), were used to determined the following counts:

| **O** = Operators | **D** = Data Streams | **T** = Abstract data types | **LGC** =**O+D+T** |
|---|---|---|---|
| **2544** total operators | **4010** total data streams | **1003** total Data Types | **7557** |

# VI. COMPARISON OF OUR LGC AND KLOC VERSUS THE NOGUEIRA ESTIMATED LGC AND KLOC

In order to validate the Nogueira model, we used the Nogueira LGC average of 40 LGC/KLOC[15] for comparing our two project's LGC calculations. For consistency, we used the same standard that Dr. Nogueira used, non-commented source lines of code. The calculation and comparison are as follows:

## A. FIRST PROJECT

LGC calculation was LGC=841+1468+483=2792. This equates to an average of 35.4 LGC/KLOC (98958/2796=35.4 LGC/KLOC) as compared to the average of 40 LGC/KLOC determined in the Nogueira dissertation (KLOC=2792*40+150=111830). This represents a delta of 11.5% (35.4/40=.885 Therefore 1-.885=.115*100=11.5) We feel this is close in correlation and is within the margin of error of this Thesis given the constraints and uncertainty of applying this method post mortem as stated in our conclusions. We believe this validates this part of the model within the range of this Thesis' data. However, additional research is being conducted and opportunities exist to validate the Nogueira formal method beyond the scope of this paper, i.e. the, simulation work being conducted at NPS using Vite' project.

## B. SECOND PROJECT

LGC calculation was LGC=2544+4010+1003=7557. This equates to an average of 38.03 LGC (287419/7557=38.03 LGC/KLOC) as compared to the average of 40 LGC/KLOC determined in the Nogueira dissertation (KLOC=7557*40+150=302430). This represents a delta of 5.00% (38.03/40=.950 Therefore 1-.950=.050*100=5.00). We

feel this is very close in correlation and is within the range of the margin of error of this Thesis's data, and validates this part of the model.

We also noticed that the Nogueira model estimated KLOC was within 11.5% for our first project and 5% for our second project (figure 12.) Because of such a close correlation, we are recommending that future research be done to validate and refine our grep search method when extracting out Operators, Data Streams and Abstract Data Types, and then used to calculate LGC that in turn is used to estimate KLOC per Dr. Nogueira dissertation.

$$KLOC=LGC*40+150$$

**Actual KLOC vs Nogueira Estimated KLOC**

| | |
|---|---|
| 1st Project Actuals | 1st Project N Estimated |
| 2nd Project Actuals | 2nd Project N Estimated |
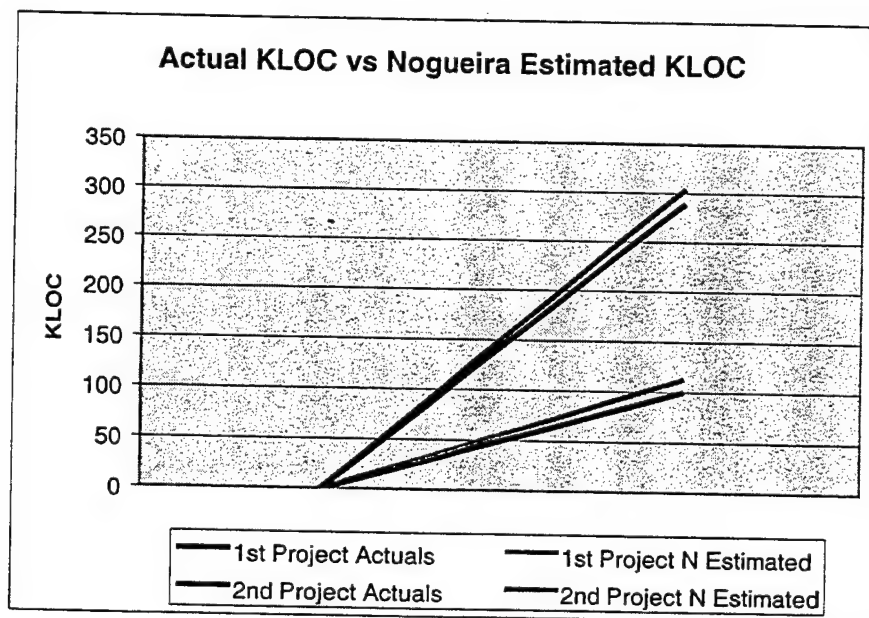
Figure 12     Actual KLOC Verses Nogueira Estimated KLOC

---

[15] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 130

## VII. THE EF (EFFICIENCY) METRIC ASSUMPTION

For our selected projects, we have assumed a "high" efficiency rating: >= 2. It was necessary to make an assumption because of the lack of data needed to analyze and calculate an EF. That kind of data was not collected at the granularity needed for our research and is due in part to the contractor not knowing what specific types of data needed to be collected to provide input to the EF metric. Further, it was reasonable for us to assume a high EF since the contractor was at SEI CMM level 3 and Requirements Volatility (RV) was low indicating stable process and high efficiency. That said, more detailed specific explanations of our assumption for each of the two projects follows:

### A. FIRST PROJECT

This high EF assumption is made since at the end of build 2, prior to the beginning of build 3, radical action was taken by the project management to stabilize and control the requirements volatility, schedule, and cost. In addition, with the deferring of the BIT Diagnostics to build 4, the design and subsequent code was less complex. Further, by this time, the Systems Engineering and Software Engineering had matured their processes via SEI CMM level 3 methodologies, and had a solid understanding of the domain thereby significantly decreasing Requirements Volatility and consequently increasing their efficiency and productivity by the start of build 3.

### B. SECOND PROJECT

This high EF assumption is made in view of the fact that the developer and customer had formed a cohesive Integrated Product Team, (IPT) which assured collaborative communications happened between Systems Engineering, Software Engineering, and the customer. This mutual flow of communications provided a boost to

efficiency by mitigating unknown changes made by either Systems or Software Engineering, thereby increasing productivity while decreasing re-work. In addition, the developer had stabilized their staffing resource, matured their processes via SEI CMM level 3 methodologies, and had a solid understanding of the domain. This is reflected in the low and consistent Requirement Volatility calculations and therefore, we believe it is logical to assume a high EF.

# VIII. PREDICTED PROBABILITY OF COMPLETION VERSUS ACTUAL TIME TO COMPLETE

## A. PREDICTED VS ACTUAL TIME TO COMPLETE FOR PROJECTS

### 1. First Project

We used our Nogueira Model MS Excel spreadsheet tool to calculate the Probability of completion curve for the projects. For consistency, we used working days, defined as 22 days per month, the same as used by the Nogueira model; we also chose to use 22 working day intervals for our chart. We then computed and plotted the Probability of Completion versus Time in days chart (figure 13.)



**First Program Probablility to Complete**
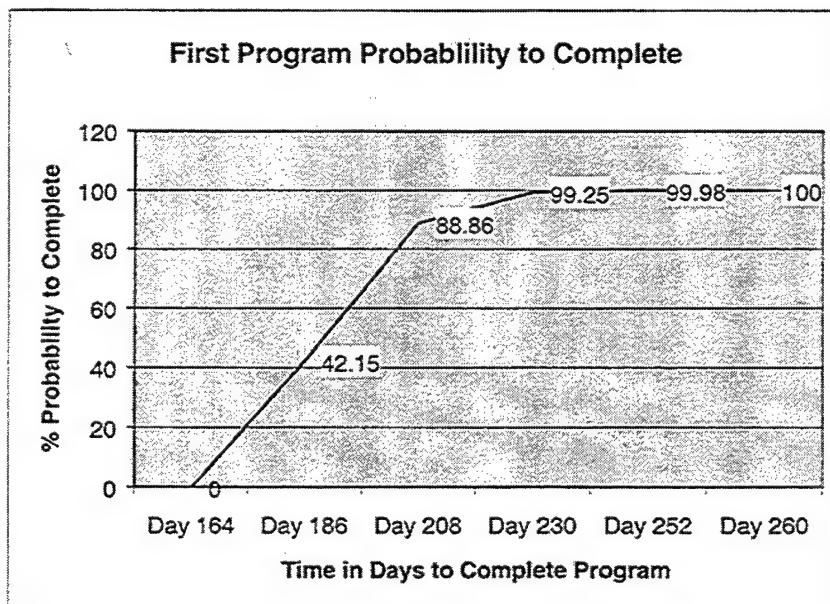
Figure 13    First Project Probability to Complete

The model predicted that the minimum time in days necessary to have a probability of completion of 100% is approximately 260 working days. When compared to the actual time it took, which was 336 working days, the model predicted completion sooner. The Model predicted 76 working days less, or a 22.6% delta.    (1 -

39

(260/336))(100)= 22.6. At this point, with 22.6% variability, we decided to investigate and see what the original estimated completion date was from project records. The original estimation was 200 working days, with the project schedule slipping 136 working days for build 3. The developer missed the original completion estimation by 40.5%. (1- (200/336))(100)=40.5. The Nogueira model missed the developer's original estimate by 23.1% (1-(200/260))(100)=23.1 (figure 14.) Does this mean that the Nogueira model is too optimistic as are most developers' estimates, or is it a better fit? This data point leaves us with an inconclusive position as to the validation of the model against the first project. It appears that there is a difference when using real projects with real data versus simulated project data, and this reflects what the real world is — unpredictable.

All Models

P(t)

1.5
1
0.5
0

0    100    200    300

Time

Model 1
Model 2
Model 3
Model 4
260.0

| Model 4 | |
|---|---|
| Efficiency | |
| Direct Labor % | 67.00% |
| Idle % | 33.00% |
| EF | 2.030 |
| Req. Volatility | |
| Birth Rate % | 2 |
| Death Rate % | 6.8 |
| RV | 8.800 |
| Complexity | |
| Operators | 841 |
| Data Streams | 1468 |
| Types | 483 |
| LGC | 2792 |
| Alpha | 1.95 |
| Beta | 28.86358635 |
| Gamma | 164.8110781 |

| Model 1 | | Model 2 | | Model 3 | | Model 4 | |
|---|---|---|---|---|---|---|---|
| Alpha | 1.95 | Alpha | 1.95 | Alpha | 1.95 | Alpha | 1.95 |
| Beta | 24.73317111 | Beta | 23.05634595 | Beta | 29.92716697 | Beta | 28.86358635 |
| Gamma | 136.0324411 | Gamma | 136.0324411 | Gamma | 155.465647 | Gamma | 164.8110781 |
| P(t) | 100.00% | P(t) | 100.00% | P(t) | 100.00% | P(t) | 100.00% |
| t (days) | 260.0 | t (days) | 260.0 | t (days) | 260.0 | t (days) | 260.0 |
| months | 12 | months | 12 | months | 12 | months | 12 |

Model 1*

Used when RV is small.

*CX >= 600 LGC
*RV <= 67%

Model 2*

Considers the three factors, but neglects combined effects of (EF & RV)

Model 3*

Considers the three factors and combines their effects

Model 4**

Model used for any range of complexity and requireme volitility, and considers all factors, with their combined effects.

**A 0 LGC will take 0 days
**Alpha, Beta, Gamma > 0
**If RV increases then P(t) de
**If CX increases then P(t) de
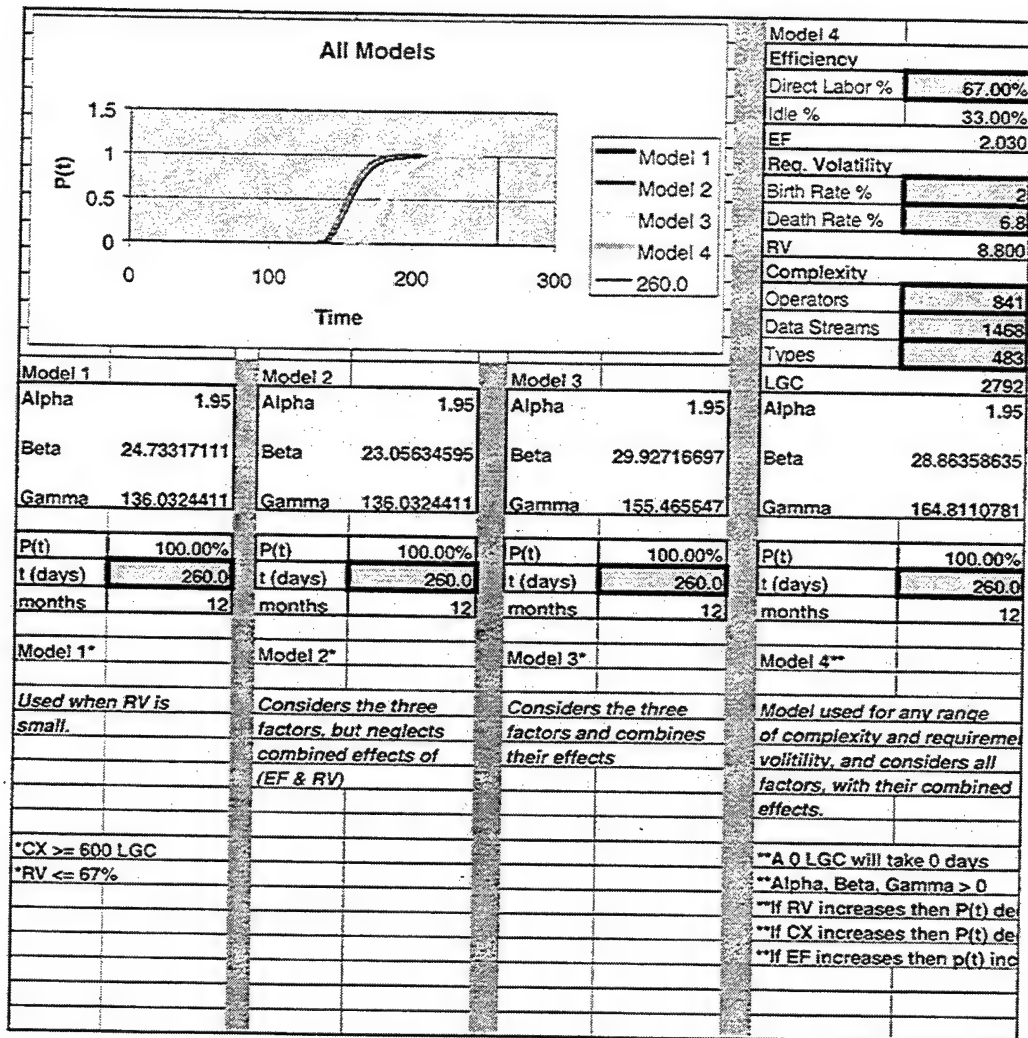**If EF increases then p(t) inc

Figure 14    First Project Model Predicted Completion

## 2.    Second Project

We used the Nogueira Model MS Excel spreadsheet tool to calculate the Probability of completion curve for Build 2 using; BR=2.59, DR=3.04, RV=5.63, O=2544, D=4010, T=1003. The model predicted Impossible (figure 15.)

| All Models | | | Model 4 | |
|---|---|---|---|---|
| | | | Efficiency | |
| | | | Direct Labor % | 67.00% |
| | | | Idle % | 33.00% |
| | | | EF | 2.030 |
| | | | Req. Volatility | |
| | | | Birth Rate % | 2 |
| | | | Death Rate % | 6.8 |
| | | | RV | 8.800 |
| | | | Complexity | |
| | | | Operators | 2544 |
| | | | Data Streams | 4010 |
| | | | Types | 1003 |

| Model 1 | | Model 2 | | Model 3 | | LGC | 7557 |
|---|---|---|---|---|---|---|---|
| Alpha | 1.95 | Alpha | 1.95 | Alpha | 1.95 | Alpha | 1.95 |
| Beta | 40.34599958 | Beta | 37.61067757 | Beta | 48.81870831 | Beta | 46.80801949 |
| Gamma | 221.9029977 | Gamma | 221.9029977 | Gamma | 253.6034259 | Gamma | 257.2737913 |
| P(t) | Impossible | P(t) | Impossible | P(t) | Impossible | P(t) | Impossible |
| t (days) | 68.0 | t (days) | 68.0 | t (days) | 68.0 | t (days) | 68.0 |
| months | 3 | months | 3 | months | 3 | months | 3 |

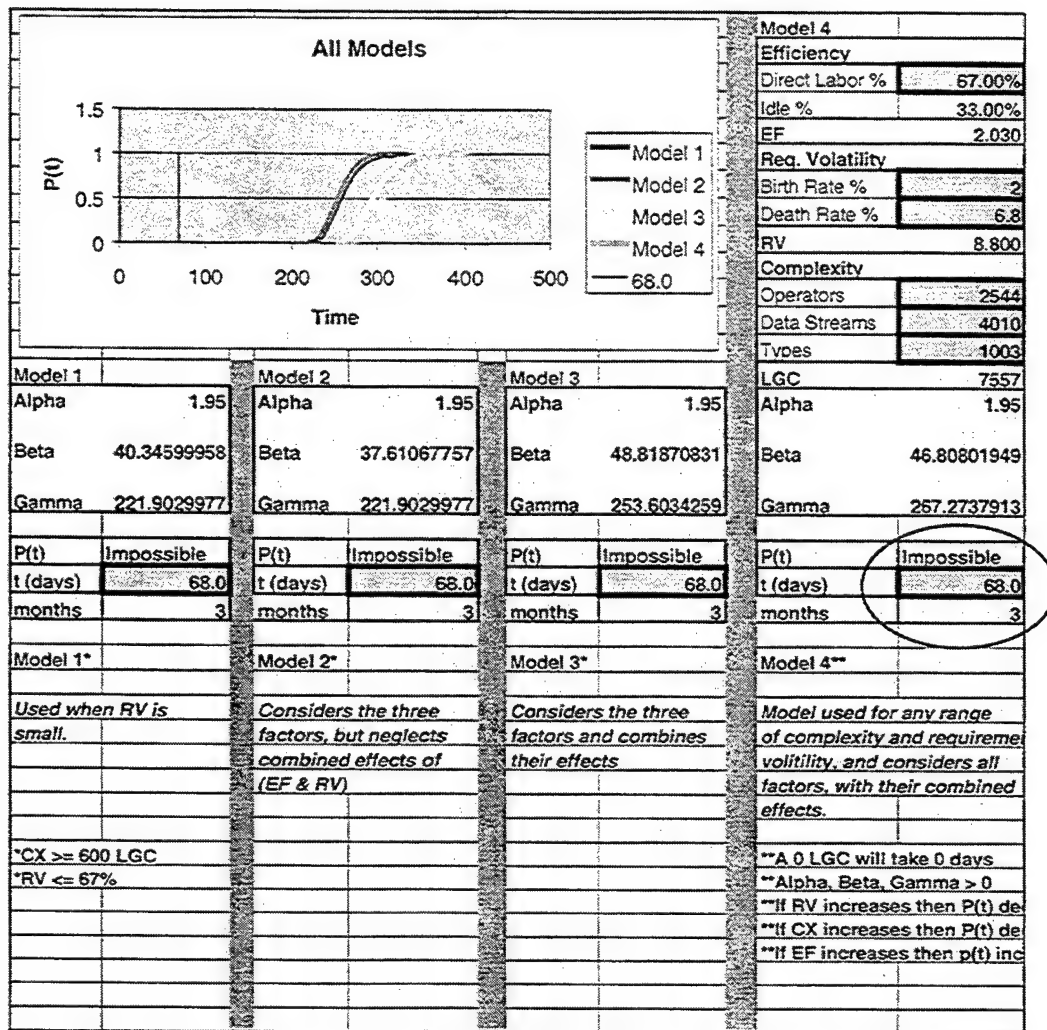| Model 1* | Model 2* | Model 3* | Model 4** |
|---|---|---|---|
| Used when RV is small. | Considers the three factors, but neglects combined effects of (EF & RV) | Considers the three factors and combines their effects | Model used for any range of complexity and requireme volitility, and considers all factors, with their combined effects. |
| *CX >= 600 LGC | | | **A 0 LGC will take 0 days |
| *RV <= 67% | | | **Alpha, Beta, Gamma > 0 |
| | | | **If RV increases then P(t) de |
| | | | **If CX increases then P(t) de |
| | | | **If EF increases then p(t) inc |

Figure 15    Second Project Model Predicted Completion

Actual time for build 2 took from 4/24/00 until 7/10/00 or 68 working days at 22 working days a month. We believe this inconsistency is due primarily because the O, D, and T counts are based on all six CSCIs, as discussed previously in III.C.1. Candidate project two.    Core functionality on three CSCIs, Fire Control, (FC), System Manager Software, (SMS), and Operating Environment, (OE) had been developed and validated. The builds during this period, involved addition of functionality to the following CSCIs: Command and Communications (C2), and Soldier Machine Interface (SMI), Vehicle Diagnostics Management System, (VDMS). That is, build 2 was modifying only a

portion of the total software system code, but the O, D, and T data gives a view of all six CSCIs combined. It was not broken down into separate CSCIs, nor does it, post mortem, identify the code that was being worked in a previous software release. We cannot fault the developer for not collecting metrics for research concepts that they are not aware of, nor do we believe that this type of data collection is a requirement of CMM level 3.

A finding of this research is the need to adjust the CX when applying the Nogueira model to evolved projects that are developing or enhancing only a portion of their CSCIs. And for accurate counts of the O, D, and T data, the CSCIs must be segmented such that you can do a LGC calculation for only the code that is being developed. This project did not utilize a lower case tool such as Rational Rose. We believe use of such a tool is essential when attempting to apply the Nogueira formal model, as it provides the capability to collect detailed information, over the software development lifecycle, that can later be extracted and used for input to the Nogueira model metrics.

THIS PAGE LEFT INTENTIONALLY BLANK

## IX.   THE COMPARISON OF KLOC – ACTUALS VERSUS NOGUEIRA ESTIMATED

### A.   FIRST PROJECT

In order to validate the Nogueira model, we used the Nogueira KLOC estimation algorithm16 for converting our LGC calculations into KLOC.  The conversion is as follows:  KLOC=(40(LGC)+150)/1000.   Therefore KLOC for our first project is calculated by ((40*2792)+150)/1000=111.8KLOC.  When compared to the actual KLOC counts of our project, we observed a difference of 11.5%.  98958 (actual KLOC)/111830 (Nogueira estimation)=.885     1-.885=.115  or translated, implies a difference of 11.5%. We believe this is suitable for a rough comparison as our LGC count, that directly impacts the Nogueira method, was determined in a manual data collected manner and so subject to more variability than if done by automated methods.  We believe that if given an absolute way to count Operators, Data Streams and Data types in our projects, the comparison would have correlated even closer.  As such, for the purposes of our Thesis, this more or less validates this part of the model.

### B.   SECOND PROJECT

In order to validate the Nogueira model, we used the Nogueira KLOC estimation algorithm for converting our LGC calculations into KLOC.  The conversion is as follows:

KLOC=(40(LGC)+150)/1000.   Therefore KLOC for our second project is calculated by ((40*7557)+150)/1000=KLOC, or 302.43.  When compared to the actual KLOC counts of our project, we observed a difference of 5%.   287419 (actual KLOC)/302430 (Nogueira estimation)= 1-.950=.050 or translated, implies a difference of

45

5%. We believe this is good correlation for a comparison as our LGC count, that directly impacts the Nogueira method, was determined by manual means of collection and not automated as stated previously. As such, for the purposes of our Thesis, it validates this part of the model within our data's range and margin of error.

---

[16] J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," <u>Software Engineering</u>

# X.    CONCLUSIONS

We found the Bi-dimensional plot a profoundly simple and useful tool to visualize the effects of volatility to assess and verify the health of a project and the effects of project activity. We believe the Bi-dimensional plot has the potential to improve a project's ability to more effectively assess volatility data. We believe its use should be tempered with an understanding of the nature of variation[17], the project, and its processes. We found that on both projects it gave fair reflections of project events and processes. We believe the Bi-dimensional plot should be used as a rule of thumb tool to help ascertain the effect of project events and actions. On these projects the tool was applied manually. For it to be most effective, it should be implemented with automated CASE tools otherwise the information's timeliness would be useful only for post mortem studies such as this. Although this Bi-dimensional plot appeared to be accurate we believe it unlikely that a single control limit of 10% for Birth Rate and Death Rate defining growing, volatile, shrinking, and stable requirements will adequately represent all software development projects. Opportunities for future research would be to apply the Bi-dimensional plot to additional projects and software releases to further validate and develop the Bi-dimensional plot control limits that define the growing, volatile, shrinking, and stable regions of the graph. Families of Bi-dimensional plots with control limits for various project types could be developed or a methodology to establish control limits for specific applications could be developed.

We believe a pilot application of the Nogueira model to real projects to demonstrate and validate its risk management capabilities is the next logical progression.

---

This effort demonstrated that the Nogueira model can be applied to real projects, in various development phases, and correlated with actual project results.

This effort demonstrated that the Nogueira model couldn't be applied to an evolved project where the development involved modifying only some of the products CSCI's and modules. For calculating LGC we only had visibility of the entire product. We were not able to isolate and count the CSCI being modified. We were only able to count O, D, & T for the entire product, which gave us an erroneous prediction of "Impossible" in a time frame where we had accomplished a build. This resulted in an inability to make a prediction with any confidence for the predicted probability of completion utilizing the Nogueira model. We believe this is due to the data available not being sufficient; this is not a failing of the developer or the Nogueira model; it does present opportunities to develop what must be in place to successfully apply the Nogueira model.

From this we have learned that applying the Nogueira formal model to evolved systems poses additional challenges. Adequate metrics may not be available, visibility and segmentation of the code product may not exist to allow counting only the modules that were modified.

We believe the Nogueira formal method is at a level corresponding to a CMM level 4 organization. Applying the Nogueira method to organizations below CMM level 4 will likely involve introducing new methods and maturing existing processes and metrics gathering thereby injecting additional overhead in the form of a learning curve.

---

[17] W. Edward Deming, Out of the Crisis (MIT Press Nov 1989) 309

We believe that attempting to apply the Nogueira method to organizations below CMM level 3 is inadvisable due to the discipline of continuously collecting and using metrics, (processes are measured and controlled), not being enforced and mandatory until a developer is at CMM Level 4.

A lesson learned from Thesis effort is that we are proposing to modify the project selection criteria to the following:

- The projects are DoD development projects utilizing the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) with a minimum of level 3, preferably level 4.

- Preferably the projects are early in their development. Evolved systems present challenges calculating LGC when only a portion of the product is being developed and modified. This may be less of a problem if metrics can be applied to design/modeling notations rather than code and if previous versions are available so that differences can be calculated.

- CASE tools are used to facilitate the implementation of automated scripts to give counts of Operators (O), Data Streams (D), and Abstract Data Types (T). For evolved systems the source code should be segmented to isolate the code modules being modified to count O, D, and T.

- Volatility data needs to be collected consistently and uniformly.

- The projects utilize the lifecycle phases of an evolutionary development.

- The project uses Object-Oriented Methodology (OOM)

- The project is Coded in Ada

- The software is real-time embedded

We found that applying the Nogueira Model post mortem, on projects, not developed with CAPS or PSDL[18], presented intellectual dilemmas that impart a level of uncertainty as to the completeness and accuracy of the data we collected to calculate LGC. Because the Nogueira Model introduces the concept of LGC, one cannot expect that projects developed without knowledge of it would do all that is possible to capture all of the data to calculate LGC.

From this application of the Nogueira Model, post mortem to projects, we have concluded that it must be recognized that the results have an undetermined level of uncertainty because we cannot ascertain post mortem, all of the streams or the intellectual nuances the developers created, especially in handcrafted code. This might change in the future as advances are made in auto code generation techniques, similar to those in CAPS.

We found this research project proved to be interesting as well as valuable in identifying unexpected model effects in areas not previously explored. Preferably, application of this model will be to projects at the beginning where the project can be prepared to more completely capture the necessary data to calculate LGC and estimate uncertainty more accurately. Further, since the model has had very limited exposure to real projects, we suggest it be distributed to the software engineering community at large, in essence, private industry and within Government at entities like NASA, and other DoD components for their use and feedback.

When we had sufficient data we found that the Nogueira model correlated well in some areas and not so well in other areas. It seemed to correlate well, within a 10%

---

[18] Computer Aided Prototyping System, Naval Postgraduate School, Luqi, Berzins, Shing

variability range when compared to average LGCs per KLOC. It also correlated well within a 10% variability when compared to LGC calculated and average LGC per the Nogueira dissertation.

Unfortunately when we applied the model to our projects the end results were inconclusive as to the validation of the model. Our variance from the model by approximately 22.6% to 23.1% put us in a position of indetermination as to how well the model will work in the real world.

Therefore in conclusion, the application of this model must be planned and applied to projects and organizations that have the sophistication and metrics available to accurately assess its value. Additional applications of this model, post mortem, could help evolve and refine the model's application criteria and methodology, and further validate its use in software development.

THIS PAGE LEFT INTENTIONALLY BLANK

# XI. FUTURE RESEARCH

## A. UNIX GREP METHOD

It appears that the method we created, the "Unix Grep search method", is a viable method for use in extracting and counting Operators, Data Streams and Abstract Data Types from source code when a tool such as PSDL and CAPS is not available or used on a project. This method can be used on any project hosted on a UNIX operating system. It can be customized for any implementation language used on a project simply by making a determination of what entities in the implementation language represent Operators, Data Streams and Abstract Data Types and then placed in the grep search string.

From our limited research of just two software projects, we were able to get very good correlations from 5 to 11.5% when comparing actual KLOC to that estimated by the Nogueira model using input from our grep search method. However, further research using this method would be needed to prove out our supposition.

## B. SYNTACTIC WORD SEARCH AND AUTOMATION

Using the grep search method and the appropriate syntax for an implementation, automation of the search script could be accomplished. This would, if proved out, create a fast and efficient way to count and total Operators, Data Streams and Abstract Data Types from source code for automatic feed into the Nogueira model metrics to calculate LGC. This method could be directly linked into our Norgueira model MS Excel spreadsheet tool. We believe this would be very useful, and a topic of future research.

## C.     BI-DIMENSIONAL PLOTS

Opportunities for future research would be to apply the Bi-dimensional plot to additional projects and software releases to further validate and develop the Bi-dimensional plot control limits that define the growing, volatile, shrinking, and stable regions of the graph. Families of Bi-dimensional plots with control limits for various project types could be developed or a methodology to establish control limits for specific applications could be developed. Additionally, future work with the bi-dimensional plot could help demonstrate and make generally known the benefits of stabilizing the core product.

# APPENDIX A – ACRONYMS AND ABBREVIATIONS

| | |
|---|---|
| BIT | Built In Test |
| BR | Birth Rate |
| C2 | Command and Communications |
| CAPS | Computer Aided Prototyping System |
| CASE | Computer Aided Software Engineering |
| CMM | Capability Maturity Model |
| CR | Change Rate |
| CSCI | Computer Software Configuration Item |
| CX | Complexity |
| D | Data Streams |
| DoD | Department of Defense |
| DR | Death Rate |
| EF | Efficiency Factor |
| ERB | Engineering Review Board |
| FC | Fire Control |
| IPT | Integrated Product Team |
| KLOC | Thousand (K) Source Lines of Code |
| LGC | Large Granularity Complexity |
| NASA | National Aviation and Space Administration |
| NPS | Naval Postgraduate School |
| O | Operators |
| OE | Operating Environment |
| OOM | Object-Oriented Methodology |
| PCR | Problem Change Requests |
| PSDL | Prototype System Description Language |
| RTM | Requirements Traceability Matrix |
| RV | Requirements Volatility |
| SEI | Software Engineering Institute |
| SMI | Soldier Machine Interface |
| SMS | System Manager Software |
| T | Types (Abstract Data Types) |
| VDMS | Vehicle Diagnostics Management System |

THE PAGE INTENTIONALLY LEFT BLANK

# LIST OF REFERENCES

1. Abdel-Hamid, T. *Software Project Dynamics: An Integrated Approach.* Prentice Hall, 1991.

2. ANSI/IEEE *Standard Glossary of Software Engineering Terminology.* STD-729-1991.

3. Albrecht, A. and Gaffney, J. *Software Function Source Lines of Code and Development Effort prediction.* IEEE Transactions on Software Engineering, SE-9, 1983.

4. Berzins, V. and Luqi. *Software Engineering with Abstractions.* Addison-Wesley, 1990.

5. Boehm, B. *A Spiral Model of Software Development and Enhancement. Computer.* May, 1988.

6. Boehm, B. *Software Risk Management: Principles and Practices.* IEEE Software, January, 1991.

7. Boehm, B. et al. Software Cost Estimation with COCOMO II. Prentice Hall, 2000.

8. Brooks, F. *The Mythical Man-Month.* Datamation. December. 1974.

9. Charette, R., Adams, K., & White, M. *Managing Risk in Software Maintenance.* IEEE Software, May-June, 1997.

10. Gemmer. *Risk Management: Moving Beyond Process.* Computer Vol. 30 Issue 5. May, 1997.

11. Gilb, T. *Principles of Software Engineering Management.* Addison-Wesley 1988.

12. Harn, M., Berzins, V. and Luqi. *Software Evolution via Reusable Architecture.* Proceedings of 1999 IEEE Conference and Workshop on Engineering of Computer-Based Systems. Nashville, Tennessee. March, 1999.

13. Humphrey, W. et al. *A Method for Assessing the Software Capability of Contractors.* CMU/SEI-87-TR-23. 1987.

14. Humphrey, W. *Managing the Software Process.* Addison-Wesley, 1989.

15. Jones, Capers. *By Popular Demand: Software Estimating Rules of Thumb.* Computer, March 1996.

16. Luqi and Ketabchi, M. *A Computer-Aided Prototyping System.* IEEE Software. March, 1988.

17. Luqi and Berzins, V. *Rapidly Prototyping Real-Time Systems.* IEEE Software. September, 1988.

18. Luqi. Software Evolution *Through Rapid Prototyping.* IEEE Computer. May, 1989.

19. Luqi. *A Graph Model for Software Evolution.* IEEE Transactions on Software Engineering. Vol. 16 No. 8. August, 1990.

20. Nogueira, J.C., Luqi, and Berzins, V. *A Formal Risk Assessment Model for Software Evolution.* SEKE 2000. Chicago, July 2000.

21. Nogueira, J.C., Luqi, and Bhattacharya, S. *A Risk Assessment Model for Software Prototyping Projects.* IEEE Workshop on Rapid System Prototyping RSP 2000. Paris, June 2000.

22. Nogueira, J.C., Luqi, , Berzins, V., and Nada, N. *A Formal Risk Assessment Model for Software Evolution.* ICSE 2000. Limerick, June 2000.

23. Nogueira, J.C., Luqi, and Berzins, V. *Risk Assessment in Software Requirement Engineering.* IDTP 2000. Dallas, June 2000.

24. Putnam, L. *Software Cost Estimating and Life-cycle Control: Getting the Software Numbers.* IEEE Computer Society Press. 1980.

25. Putnam, L. and Myers, W. *Measures for Excellence. Reliable Software On Time Within Budget.* Yourdon Press, 1992.

26. Software Engineering Institute. *Software Risk Management.* Technical Report CMU/SEI-96-TR-012. June, 1996.

27. USAF. *Software Risk Abatement.* ASFC/AFLC pamphlet 800-45, US Air Force Systems Command. Andrews AFB. 1988.

# BIBLIOGRAPHY

J.C. Nogueira, "A Formal Model for Risk Assessment in Software Projects," Software Engineering Department Naval Postgraduate School Ph.D. Dissertation, September 2000: 128

Prime Contractor, Software Engineering Technical Reports, 1997-2000

Luqi et al., Computer Aided Prototyping System (CAPS) for Heterogeneous Systems Development and Integration (Naval Postgraduate School, 2001) 1-13

Carnegie Mellon University, Software Engineering Institute, The Capability Maturity Model (Addison Wesley, 1994) 32-35.

W. Edward Deming, Out of the Crisis (MIT Press, 1989) 309

Piirainen, R., Tank Automotive Research and Development Command, WCFO Weekly Reports, TARDEC, Santa Clara CA 95050, piirainr@tacom.army.mil

Johnson, C., Defense Contract Management Agency, Senior Computer Specialist, Software Technical Reports, DCM – San Francisco, Sunnyvale CA 94089, csjohnson@dcmdw.dcma.mil

DEPARTMENT OF THE AIR FORCE Software Technology Support Center (STSC), Guidelines for successful acquisition and management of software intensive systems, Vol. 1, Hill AFB, Utah. 1999.

THIS PAGE INTENTIONALLY LEFT BLANK

# INITIAL DISTRIBUTION LIST